Supreme Court, U. S.
FILED
AUG 4 1980
MICHAEL RODAK, JR., CLERK

Nos. 79-855 and 79-1112

IN THE

Supreme Court of the United States

October Term, 1979

No. 79-855.

SIDNEY A. DIAMOND, Commissioner of Patents and Trademarks, *Petitioner,*

v.

JOHN J. BRADLEY and BENJAMIN S. FRANKLIN, *Respondents.*

No. 79-1112.

SIDNEY A. DIAMOND, Commissioner of Patents and Trademarks, *Petitioner,*

v.

JAMES R. DIEHR, II and THEODORE A. LUTTON, *Respondents.*

On Writs of Certiorari to the United States Court of Customs and Patent Appeals.

BRIEF AMICI CURIAE FOR APPLIED DATA RESEARCH, INC. (ADR) and WHITLOW COMPUTER SYSTEMS INC. (WHITLOW).

MORTON C. JACOBS,
Suite 1420,
1420 Walnut Street,
Philadelphia, Pennsylvania 19102
Attorney for Amici ADR and Whitlow.

International Printing Co., 711 So. 50th St., Phila., Pa. 19143 — Tel. (215) 727-8711

QUESTIONS PRESENTED.

A. Questions of Law.

1a. Whether subject-matter patentability of a machine under 35 USC 101 can be different for different modes of machine construction.

b. Whether a computerized machine or industrial process that is patentable subject matter under 35 USC 101 when constructed with a hardware program (wired circuits) would also be patentable subject matter when constructed with a stored computer program (i.e., firmware or software).

2. Whether the Supreme Court is an appropriate body for resolving the complex economic and public policy issues underlying the Patent Office request that the Court exclude from subject-matter patentability a computerized machine or industrial process implemented by a stored computer program where the Patent Office has already been to the Congress, and has failed to get such exclusionary legislation.

B. Questions of Fact.

1. Whether a general-purpose computer containing a novel stored program (firmware or software) is converted by the stored program into a different machine from the general-purpose computer alone.

2. Whether those skilled in the art commonly construct machines alternatively by hardware programs completely or by a general-purpose computer with a stored program (firmware or software).

C. Question of Fact and Law.

Whether a computerized machine built with a stored program differs from the same machine built with a hardware program in any respect that would be material to its subject-matter patentability.

TABLE OF CONTENTS.

	Page
QUESTIONS PRESENTED	i
A. Questions of Law	i
B. Questions of Fact	i
C. Question of Fact and Law	i
INTEREST OF AMICI CURIAE	1
SUMMARY OF ARGUMENT	6
ARGUMENT	8
I. The Patent Office in Effect Calls for a New Rule of Law That Would Predicate Subject-Matter Patentability of a Computerized Machine on Its Mode of Construction; Machines Built With a Stored Program (Firmware or Software) Would Not Be Patentable, Even Though Those Built With Hardware Programs Are	8
II. Machines Built With Stored Programs; the Myth of the Non-Machine	10
A. What Is a Machine?	10
B. Why No Factual Record of Building Machines Using the Restructuring Principle of Stored Programs	13
C. The Myth of the Non-Machine Dispelled	14
III. These Computer-Program Appeals Are for Administrative Convenience, Not to Resolve a Dispute as to Law or Fact	17
CONCLUSION	22
APPENDIX A:	
Excerpts From Brief Amicus Curiae for the Association of Data Processing Service Organizations (ADAPSO)	A1
APPENDIX B:	
Letter Dated January 26, 1979 From Jacobs to Baruch	A21
Letter Dated April 30, 1979 From Jacobs to Baruch	A24
Letter Dated May 28, 1979 From Baruch to Jacobs	A29

TABLE OF CITATIONS.

Cases: Page

Burr v. Duryee, 68 U. S. 531 ..... 10, 11
Dann v. Chatfield, No. 76-1559, cert. dnd (1977) 8
Dann v. Johnston, 425 U. S. 219 (1976) 7, 8
Dann v. Johnston, No. 74-1033 ..... 12, 13, 14, 15
Dann v. Noll, No. 76-1558, cert. dnd. — U. S. — (1977) 8
Data Cash Systems, Inc. v. JS&A Group, Inc., et al., 480 F. Supp. 1063 (DC, Ill. 1979) 21
Diamond v. Chakrabarty, — U. S. — (June 1980) 17, 20
Diamond v. Sherwood, pet. for cert. pending 8
Gottschalk v. Benson, 409 U. S. 63 (1972) 8, 20
In re Johnson and Parrack, 589 F. 2d 1070 18
In re Johnston, 502 F. 2d 765 (1974) 5
Parker v. Flook, 437 U. S. 584 (1978) 7, 8, 20

Statutes:

35 USC:
- Generally 20
- 101 ..... i, 6, 9, 10
- 102 9
- 103 9
- 106 19
- 131 19
- 135 9

United States Constitution:

Article I, Section 8 21

TABLE OF CITATIONS (Continued).

Authorities:	Page
Computer Software Management U. S. Dept. of Commerce; NBS Special Publication 500-11, July 1977	4, 5
Hearings on H. R. 5924 Before Subcomm. No. 3 of Comm. on Judic., April 20, 1967	19
Minsky, M. L., Computation: Finite and Infinite Machines, 1967, p. 201, f.n.	9, 11
2 New Encyclopedia Brittanica: Macropaedia, 505	11
Robinson on Patents (1890), Vol. 1, p. 257, § 173	10, 11
Science, Am. Ass. Adv. Science, Vol. 195, No. 4283, 1977	3
Technology Review, M.I.T., Vol. 72, pp. 18-19, May 1970	15

Nos. 79-855 and 79-1112

IN THE
SUPREME COURT OF THE UNITED STATES

OCTOBER TERM, 1979

No. 79-855
SIDNEY A. DIAMOND, Commissioner of Patents and Trademarks, *Petitioner*,
v.
JOHN J. BRADLEY and BENJAMIN S. FRANKLIN, *Respondents*.

No. 79-1112
SIDNEY A. DIAMOND, Commissioner of Patents and Trademarks, *Petitioner*,
v.
JAMES R. DIEHR, II and THEODORE A. LUTTON *Respondents*.

ON WRITS OF CERTIORARI TO THE UNITED STATES COURT OF CUSTOMS AND PATENT APPEALS.

BRIEF AMICI CURIAE FOR APPLIED DATA RESEARCH, INC. (ADR) AND WHITLOW COMPUTER SYSTEMS INC. (WHITLOW)

Applied Data Research, Inc (ADR) and Whitlow Computer Systems Inc. file this brief *amici curiae* in opposition to Petitioner, together with letters of consent of the parties in accordance with the Rules of the Court.

INTEREST OF AMICI CURIAE.

ADR was incorporated in 1959 and since then has been a small business engaged in the computer field in the

marketing of various computer services, and proprietary software products. ADR presently markets more than a dozen software products, and has been awarded two software patents, one of which was the first patent issued for a software product on the market.[1] This first patent was issued to ADR's AUTOFLOW System, which has been one of the most successful standardized software products on the market. ADR also was granted a software patent for a control system on the complex monitoring of usage of telephone lines for the improvement of telephone service.[2]

Whitlow Computer Systems Inc. was incorporated in 1969, and since then has been a small business engaged in the computer field in developing, marketing and servicing of a proprietary software product known as SyncSort. This SyncSort product provides a computerized sorting process for which a patent was recently issued.[3] This innovative sorting process has overcome important machine obstacles to the efficient performance of sorting, which is a major part of most business data processing. SyncSort reduces the amount of computer memory that is required and the amount of computer operating time, and has been recognized by wide acceptance in the computer industry.

Though small companies, both ADR and WHITLOW have successfully competed with a giant hardware man-

1. U. S. Patent No. 3,533,086 of Martin A. Goetz, issued October 6, 1970 for "Automatic System for Constructing and Recording Display Charts". The other ADR patent is U. S. No. 3,380,029 of Martin A. Goetz, issued April 23, 1968 for "Sorting System". The issuance of this patent received considerable publicity as a so-called "first software patent".

2. U. S. Patent No. 4,066,843 of David Waks, et al., issued January 3, 1978 for "Telephone Circuit Monitoring System". ADR has sold the business of this product line.

3. U. S. Patent No. 4,210,961 of Duane L. Whitlow and Azra Sasson, issued July 1, 1980 for "Sorting System".

ufacturer such as IBM in the technical development and marketing of their software products.

ADR and WHITLOW have the interest of advising the Court as to the technological basis for the granting of software patents such as those they have received.[4] Software patents are patents on special-purpose machine systems that, because of technological and economic reasons, are constructed by means of stored programs (i.e., software) and a general-purpose computer.

As software manufacturers, *amici's* experience agrees with the views expressed by Dr. Ruth M. Davis, former Director of the Center for Computer Services and Technology of the National Bureau of Standards (in an address at the 1977 ADAPSO Management Conference):

> "Software gives us the capability of using computers creatively, flexibly and economically. It is a product that can be designed to benefit users innovatively and uniquely."

However, Dr. Davis also concluded (Science, Am. Ass. Adv. Science, Vol. 195, No. 4283, 1977):

> "The first 25-year history of software has not been marked with the successes, the advances, or the decreasing costs associated with computer equipment or hardware . . . software is the most unsafe, the least understood, and the most expensive component of total computer system costs. Software development costs are almost 10 per cent of total system costs. . . . The really useful and exciting advances in computing will probably only proceed at the same pace as advances in software engineering. And this will be distressingly so."

---

4. ADR and WHITLOW have no interest in and take no position with respect to the patentability of the particular invention of respondents' patent applications.

Dr. Davis testified before Congress on the "crucial computer problem" confronting our nation:

> "The crippling problems are lining up in the software rather than in the hardware field. In particular, they fall under what we call the application-independent problem category . . . these are the software problems which are common to many applications." (Testimony given before the Government Activity Subcommittee, House Committee on Government Operations, May 16, 1972).

This software is the type universally used, and it strongly affects machine efficiency; therefore, a hardware manufacturer may not always have the incentive to improve the software, where the "crippling problems have been". Dr. Davis has also testified to "The small, mostly 'young and struggling', companies who 'surely possess excellent technical innovative capabilities' and 'the principal technical capability . . . to work . . . on our critical computer problems" (Committee Testimony, *supra*, p. 15). The Assistant Secretary of Commerce for Science and Technology in 1979 summarized his agreement (see attached App. B, A29) as follows:

> "There is a need for software innovation in the computer industry and there is a small business potential to innovate in the software field. We agree too that the patent system is important in stimulating technological innovation."

The patent incentive is both a form of recognition and a major aid in obtaining the necessary financing without which these small companies face the prospect of shriveling and dying. The need of the incentive of patents for software is as least as great as that of the incentive available for hardware, because: "Today, providing computer software involves greater . . . risk than providing computer . . . hardware. . . ." (Computer Software Management.

U. S. Dept. of Commerce; NBS Special Publication 500-11, July 1977, p. 1).

Honeywell, the respondent herein, has divergent interests in patent protection for stored-programs, which take two forms, namely software and firmware. Honeywell is opposed to patents for software-built machines (see Br. Opp. 2).[5] In his dissenting opinion in *In re Johnston*, 502 F. 2d 765, 774 (1974), Rich, J. noted the economic interest of the hardware manufacturers in opposing such patents:

> ". . . the collective forces of major hardware (i.e., computer) manufacturers and their representative associations . . . for economic reasons, did not want patents granted on *programs* for their machines."

The Brief *amicus curiae* for National Semiconductor Corporation misstates (at p. 2) this interest of the hardware manufacturers who have consistently opposed patents for software-implemented inventions, as does National Semiconductor. Moreover, National Semiconductor's brief takes note of the "conversion to hardware of what was previously deemed software" (at p. 3), and indicates that its interests are to be free of patents for machine inventions (initially implemented in software), so that it can undertake "conversion to hardware" without restriction by a software inventor's patent.

The real strides forward in computer development have occurred in hardware, an area fully protected by the patent laws. They have not occurred in software, "the most expensive component of total computer system costs." (Dr. Davis, *supra*). It would seem evident that the profit-making corporations that oppose patents for the software technology do so, because it would be to their profit advantage.

---

5. Brief in Opposition of respondent Bradley; references to briefs are to those in the *Bradley* case, except where otherwise indicated.

SUMMARY OF ARGUMENT.

The writ of certiorari should be dismissed as improvidently granted. For the Court to resolve what the Commissioner of Patents presents as the key issues in this case, it would have to undertake a thorough inquiry into the complex technological facts of the construction of computerized machines. But the record does not provide the factual basis which would enable the Court to make such an inquiry, and to review the fact finding by the C. C. P. A. into the nature of this technology, which the C. C. P. A. resolved in prior cases. The Supreme Court is not the appropriate forum for initial fact finding and though eight computer program cases have been brought to the Court by the Patent Office on certiorari, none has contained a factual record of the nature of this technology.

Nominally the question presented by the Patent Office is whether computer programs are patentable subject matter under 35 USC 101. However, the real issue, not acknowledged by the Patent Office, is that by implication the Patent Office calls for a new rule that would predicate subject-matter patentability of a computerized machine on its mode of construction: While machines built with hardware programs are patentable, those built with stored programs (firmware or software) would not be patentable. There is no basis in law to deny patentability to a machine having one mode of construction as against another. That this issue of law is the underlying one in this case can be appreciated from a basic technological fact that a stored program restructures a general purpose computer into a different machine. That technological fact has been found by the C. C. P. A. and the Board of Appeals in predecessor cases. The underlying position of the Patent Office in this case is to ignore these predecessor findings of fact, and to assert that all computer pro-

grams are in the form of algorithms, and whether directed to mathematics or not, they are unpatentable under this Court's prior rulings in *Benson* and *Flook*. The C. C. P. A. and the Board have both found the basic technological fact; their finding should be given special weight by the Court, since each has special credentials of technological expertise, and has found in favor of the restructuring principle.

The background of these cases shows a history of the Patent Office seeking to avoid the administrative difficulties of the examination of computerized machines built with stored programs. The Patent Office unsuccessfully sought legislation from Congress to exclude such inventions from subject matter patentability. Thereafter, the then Commissioner of Patents announced a policy against the issuance of software patents to discourage the filing of such patent applications. The predecessor Commissioner decided not to pursue the practice of appeals to the Court, but with the change of Patent Office administration, the current cases have been brought. The stored-program technology antedates the Patent Code; it is not a new technology requiring special consideration of Congress.

The myth that computerized machines built with stored programs are non-machines is built on *legal* argument. The facts that a computerized device built with a stored program is a machine, and that the innovative aspect of it is a machine structure become plainly apparent when a few simple illustrations are considered. Copyrights would not protect such innovation.

ARGUMENT.

I. The Patent Office in Effect Calls for a New Rule of Law That Would Predicate Subject-Matter Patentability of a Computerized Machine on Its Mode of Construction; Machines Built With a Stored Program (Firmware or Software) Would Not be Patentable, Even Though Those Built With Hardware Programs Are.

The rule proposed by the Patent Office would in its effect establish subject-matter patentability based on the way a machine is built, a false dichotomy which has no support in the Constitution, in statute, nor in case law. This rule can be understood by an example; though hypothetical, it is based upon real experiences:

An inventor demonstrates his new invention to his patent attorney with great pride; he has developed a machine for reading books out loud to the blind. The machine is composed of both a reading and talking computer. After the demonstration, the patent attorney responds:

> "What's inside the cabinet? Did you build it with software or hardware (a stored program or hardware program)? If built with a hardware program, your machine would be patentable. But if you built it with a stored program, the Patent Office would say it was merely mathematics and, therefore, unpatentable."

That state of the law is what the Patent Office has sought in each of the eight "computer program" cases[6] that it has brought since 1971. But such a rule that would set up a dichotomy in subject-matter patentability based

6. *Gottschalk v. Benson*, 409 U. S. 63 (1972); *Dann v. Johnston*, 425 U. S. 219 (1976); *Parker v. Flook*, 437 U. S. 584 (1978); *Dann v. Chatfield*, No. 76-1559, cert. dnd (1977); *Dann v. Noll*, No. 76-1558, cert. dnd. — U. S. — (1977); and besides these two cases, that of *Diamond v. Sherwood*, pet. for cert. pending.

on the mode of construction would make no sense in fact or in law. The reading and talking computer has all the attributes of utility and patentable invention whether built with hardware or a stored program. The stored-program construction is often the more powerful mode of machine construction (Minsky, M. L., Computation: Finite and Infinite Machines, 1967, p. 201, f.n.); today it is probably the only practical way to build the bookreading machine, both from the standpoint of time and cost.

To deny subject-matter patentability to the stored-program embodiment would be contrary to all these statutory requirements: (i) the inventor shall be first (35 USC 102); (ii) priority shall be determined on reduction to practice, e.g., the first successful construction and operation (35 USC 135); and (iii) "Patentability shall not be negatived by the manner in which the invention was made" (35 USC 103).

The Patent Office briefs bypass the existence, let alone patentability, of hardware programmed machines and state the Questioned Presented as follows:

> "Whether a computer program . . . is patentable subject matter under 35 USC 101." (Br. P. 2; see also Petitioner's Brief in *Diehr*, at 2, and Petition in *Sherwood*, at 2).

The term "computer program" used by the Patent Office applies only to one form of "computer program", the stored program (firmware or software). The Patent Office does not acknowledge that the computer experts tell us that hardware and stored programs serve as alternative modes of construction (see Minsky, *supra*, and attached App. A, A12-17), nor does the Patent Office acknowledge that it considers computerized machines built with hardware programs to be patentable under 35 USC 101.

It would seem clear that if the Court is to adopt the rule requested by the Patent Office that machines built

with stored programs are not patentable (but those with hardware programs are) the Court would have to deal with the following questions:

a) In what ways does a computerized machine built with a stored program differ from the same machine built with a hardware program?

b) Are any of those differences between hardware-program and stored-program machines material to the issue of subject-matter patentability?

The factual record supplied by the Patent Office does not enable the Court to deal with either of those questions. The Patent Office has not even supplied the Court with the technological facts of stored-program machines though it seeks a sweeping rule in these cases excluding them from 35 USC 101.

For that reason alone, the Patent Office case is fatally flawed, and the writs of certiorari in *Bradley* and *Diehr* should be dismissed as improvidently granted.

II. Machines Built With Stored Programs; the Myth of the Non-Machine.

A. What is a Machine?

Though we are here dealing with a class of machines that were unimaginable in the 19th century it is useful to start with the definitions of that era:

As the Court said in *Burr v. Duryee*, 68 U. S. 531, 570:

"A machine is a concrete thing; consisting of parts, or of certain devices and combination of devices. The principle of a machine is properly defined to be . . . that peculiar combination of devices which distinguishes it from other machines."

Robinson on Patents (1890), Vol. 1, p. 257, § 173, defines a machine as:

> "An instrument composed of one or more of the mechanical powers, and capable, when set in motion, of producing, by its own operation, certain predetermined physical effects. It is an artificial rule of action, receiving crude mechanical force from the motive power, and . . . transforming . . . it according to the mode established by that rule."

A renowned computer scientist and Professor of Electrical Engineering at the Massachusetts Institute of Technology has given this definition:

> "When we talk about a machine we have in mind not only (1) an object of some sort, but (2) an idea of what that object is supposed to do. . . . One expects a machine to have a significant number of 'parts' and to perform some reasonably complex operation on something, but it is difficult to capture in the same definition machines which peel apples and machines which transform coded information signals!" (Minsky, *supra*, p. 3).

The program is the "rule of action" (Robinson *supra*) or "principle" (Burr, *supra*) of a computerized machine. It determines what the machine "is supposed to do," (Minsky, *supra*), "*what* the automated system shall do and how the parts of the system must function in order to accomplish the desired result." (2 New Encyclopedia Brittanica: Macropaedia, 505, 506; see App. A, A8, f.n. 38).

A fundamental technological feature of the computer technology is the use of programs to structure the desired machine: namely, the state and configuration of the computer's electronic circuitry and other apparatus can be built with hardware programs or alternatively with stored programs. This fact of two modes of circuit configuration for building a computerized machine was admitted by

the Patent Office (in *Dann v. Johnston*, No. 74-1033, see Brief for Petitioner at 14-15) as follows:

> "The particular operations to be performed by a computer are governed by the state ['the internal molecular or electronic condition of a part'] and configuration ['the physical location and arrangement of parts'] of its electronic circuitry and other apparatus. Such circuitry can be fully built into the computer through wires and other physical elements (i.e., as part of the 'hardware'), in which case, unless it is physically rewired, the computer will ordinarily serve only a specific purpose or application.
>
> Alternatively, the computer may be designed so that the state or configuration of its circuitry can readily be changed without rewiring or other visible physical changes. Such changes are accomplished by the use of different programs, . . . (e.g., punched cards, tapes, disks) . . ."

This admission sums up the essential difference between building the rule of action (i.e., "operations to be performed") into a computer: either "through wires . . . as part of the 'hardware' . . . [or] without rewiring or other visible physical changes . . . by the use of different [stored] programs . . ." Moreover, the "configuration of its circuitry" in each case determines the rule of action. Therefore, changing that configuration changes the machine, and the circuit "configuration . . . can be readily changed [by stored] programs."

This Patent Office admission to the Court of the technological facts of computers and stored programs contradicts the factual assertion made by the Patent Office herein (Br. P. 8-9):

> "Firmware does not convert the conventional computer . . . into a new . . . machine",

and (Br. P. 21):

> "Nor does respondents' microprogram somehow transform the old hardware into a new machine . . ."

These factual assertions by the Patent Office are not supported by any factual (nor legal) authority, and they are not presented to the Court for the Court's *finding of fact.* To the contrary, they are presented to the Court for its approval *as a legal conclusion,* notwithstanding that it is contradicted by the finding of fact of the Patent Office Board of Appeals in *Ex parte Johnston* which reversed the Examiner on precisely this issue (see decision quotation in attached App. A, A3, f.n. 24).

A stored (wireless) program serves to configure the computer circuits, just as a hardware (wired) program does; and it is similar to the configuration of mechanical parts in a mechanical machine. In each case, the configuration of devices supplies the rule of action. To remove the stored program from the general-purpose computer is to remove its unique rule of action as a machine. This would be the same as untying the wires interconnecting the circuits of a hardware-program computer, or disassembling the wheels, gears and levers of a mechanically programmed machine; and conversely the same.

B. Why No Factual Record of Building Machines Using the Restructuring Principle of Stored Programs.

In none of the eight "computer program" cases has the Patent Office met its responsibility as petitioner to supply a factual record of the complex technology of computers and computer programs. Respondents on the other hand have always relied on the fact finding of the C. C. P. A., which has an established expertise in technology. Instead of giving the Court the facts on what is largely (and, *amici* submit, now entirely) an issue of technology, the prestigious Patent Office, noted for its knowledge of advanced technology, relies on legal argument

and rhetoric. When the Patent Office argues law and merely denies and disparages the facts, but does not develop a factual and explanatory record, the lawyer's old adage suggests: the technological facts are against it!

C. The Myth of the Non-Machine Dispelled.

The restructuring principle of the modern computer, whose stored program configures and controls the computer's circuits, taxes the credulity of the uninitiated. How can what seems to be a "recording" of electrical signals (like a musical record) produce a new machine? Such a question was directed from the Court during the *Johnston* oral argument (see attached App. A, A1-15 for an answer based on legal and scientific authorities). In *Johnston,* there were findings of fact accepting the restructuring principle by both tribunals below, the PTO Board of Appeals, as well as the C. C. P. A. Therefore, the Patent Office was compelled to admit in its *Johnston* brief (*supra*), that, by means of a stored program in a computer, the "configuration of its circuitry can readily be changed without rewiring or other visible physical changes."

No such acknowledgment has been made here; to the contrary, the Patent Office denies the restructuring principle (Br. P., 21):

> "Nor does respondents' microprogram somehow transform the old hardware into a new machine. . . . Still less does the recording of the microprogram in the control unit turn the combination of the programmed unit and the conventional elements of the computer into a new machine. The only machine involved here is the familiar general purpose digital computer, unchanged by respondents in any of its hardware components . . ."

The Patent Office offers no legal or scientific authority nor factual explanation for that denial of the facts, nor justification for extending it to the following analogy (Br. P. 18, f.n. 27):

> "It is as if respondents claimed a new machine from the combination of an old player piano and a piano roll inscribed with a song they had created".

The spuriousness of this analogy is readily seen, for a computer program has been used to restructure a general-purpose computer (e.g., the same computer used in the book reading machine) to make (give it the rule of action of) a musical synthesizer. The latter then operates like a player piano for receiving punch cards (piano rolls) as inputs and playing them (for a photographic illustration, see Technology Review, M.I.T., Vol. 72, pp. 18-19, May 1970, or the Appendix of Brief for Respondent in *Dann v. Johnston*, No. 74-1033). But a piano roll does not restructure an old player piano; it remains nothing more than a player piano even with the piano roll.

The restructuring principle is also easily seen in other well known examples of the use of stored programs: to build a talking computer; or a computer for recognizing (reading) printed characters; or to combine those two with additional stored programs to build the aforementioned machine for reading books out loud (for other examples, see attached App. A, A11).

The Patent Office argues that all stored programs are "necessarily . . . mathematical" (Br. P. 14):

> ". . . because a digital computer is designed solely to perform mathematical computations, any program that it can implement must necessarily be mathematical."

However, its premise is wrong, because among the most unique functions of general-purpose computers are those

of "Conditional Transfer of Control" (see App. A, A6-7). By means of these control functions, stored programs attain extraordinary flexibility and power in the operations they can perform. The variety of machines which have been, and can be, built thereby must boggle the imagination, and they are not just mathematical machines.

The Patent Office seems to contend that all innovation, in the stored program field, for some years past and forevermore, is merely a matter of coming up with an abstract idea, drawing some flow chart diagrams and giving it to a non-innovative programmer to "translate" into suitable code (Br. P. 12). The Patent Office argues, in effect, that the general-purpose computer is so extraordinary that this process can be applied to any machine that one can imagine; one need only say "there ought to be machine that . . ." and the appropriate routine tasks can be performed to build it. Needless to say, the inventor of the reading and talking computer would not recognize his own experience in such a disparaging and unrealistic description (see App. A, A17-20).

If there were any substance to this Patent Office argument that we can leave such new developments to the non-innovative programmer to "translate" into a program, it would bring to an end the need for inventions and "presage the arrival of that period when human improvements must end", as the first Commissioner of Patents, Henry L. Ellsworth, in his 1843 report to Congress (PTO Off. Info. Serv.) stated:

> "The advancement of the arts, from year to year, taxes our credulity and seems to presage the arrival of that period when human improvements must end."

Needless to say the millenium has still not arrived, the need for software innovation is still with us, and the incentives of the patent system are still required.

In *Diamond v. Chakrabarty*, — U. S. — (June 1980), the Court upheld the patentability of a living micro-organism constructed with a novel program. Certainly, since the latter is patentable subject matter, and since machines built with hardware programs are also patentable, there is no reason to exclude, from subject matter patentability, programmed machines built with stored programs.

In the absence of a factual record, neither the present nor the predecessor computer-program cases could deal with the underlying technological issues: the non-machine as against the restructuring principle of stored programs. The contentions placed before the Court suggest a game based on the story of the blind men and the elephant: "Feel this (the tail), it's a rope"; "And this (the ear), it's a fan"; and so on. Instead of the technological facts of these machines, superficial aspects are highlighted (all programs are mathematical or abstract ideas; the material is not transformed; the parts still work the way they were intended) without bringing to light the total nature and complexity of the programmed computer.

This blind-man game would quickly terminate after a simple physical demonstration to the Court—that of inserting a series of different programs into a general purpose computer to produce a corresponding series of different machines.

III. These Computer-Program Appeals Are for Administrative Convenience, Not to Resolve a Dispute as to Law or Fact.

In April 1979, representatives of two software companies (amici herein) met with the Assistant Secretary of Commerce for Science and Technology, the Honorable Jordan J. Baruch, to discuss patent protection for software (see correspondence thereon in attached App. B, A21-29).

This meeting was requested by software people following an address by Mr. Baruch to the patent bar on the Administration's patent policy and our nation's need for technological innovation. Mr. Baruch's background as a computer scientist and his administrative relation to the Patent and Trademark Office placed him in a special position to deal with this matter. Commissioner of Patents Donald W. Banner and Patent Office Solicitor Joseph F. Nakamura also attended the meeting.

Assistant Secretary Baruch wrote in summary of the meeting (App. B, A29):

> "We indeed had a very useful exchange of ideas. As you point out, we can quickly 'stipulate' to several things. There is a need for software innovation in the computer industry and there is a small business potential to innovate in the software field. We agree too that the patent system is important in stimulating techonological innovation."

The principal issue of discussion was the technological nature of computer programs. On this point, Mr. Baruch noted that, from an engineering viewpoint, choosing between software or hardware for building a computer system was essentially a matter of evaluating the relative costs (App. B, A24), that is, a matter of tradeoffs in economics and other factors of relative advantage. This was an acknowledgment of the well known technological principle of the engineering equivalence of software and hardware (see App. A, A12-17).

Thereafter, in June 1979, the Commissioner of Patents announced in an address to a patent bar conference that he would not seek certiorari to this Court from the C. C. P. A.'s reversal of the Patent Office in the computer programming case of *In re Johnson and Parrack*, 589 F. 2d 1070. Instead the Commissioner announced he would go

forward with the preparation of guidelines for the examination of inventions implemented with stored programs:

> "Before leaving the subject of computers, you may have noticed that the PTO did not petition for a writ of certiorari in the Johnson and Parrack case. On the contrary, we are at the present time giving thought to the preparation of guidelines on the patentability of computer programs." (PTO Off. Info. Serv.).

However, with the entry of judgment by the C. C. P. A. in the *Bradley* case on July 5, 1979, and the appointment of a new Commissioner of Patents (following the resignation of Commissioner Banner), a new decision was made to seek certiorari in the *Bradley* and *Diehr* cases.

This same pattern in the history of this issue has repeated for the past fourteen years. In 1966, the Patent Office issued Guidelines for Examination of Computer Programs which recognized the patentability of machines built by stored programs (see App. A, A2-3, f.n. 22, 23). Thereafter, the Patent Office sought legislation proposing 35 USC 106 (H. R. 5924 and S. 1042, 90th Cong. 1st Sess., 1967), which would have excluded a "plan of action" from patentability. This legislation failed when the President's Science Adviser discredited it (because it sought to exclude software-built machines from patentability, while granting patents for the equivalent hardware programs) (see Hearings on H. R. 5924 Before Subcomm. No. 3 of Comm. on Judic., April 20, 1967, at p. 160).

In 1968, Revised Guidelines were issued to refuse patents for stored-program inventions, based upon "policy" (i.e., not on a basis of fact or law), and in contravention of the mandate of 35 USC 131. The Commissioner of Patents stated the purpose of this "policy":

> "The present Office policy of refusing patents in this [computer program] field has acted as a strong deterrent to the filing of applications". (Commissioner

> Brenner, "The Future of Computer Programs in the U. S. Patent Office", Law of Software, 1968, p. B-12).

However, the C. C. P. A. in its appellate review of this Patent Office "policy" decision has overturned that decision, and in a line of cases (which follow this Court's ruling on the non-patentability of mathematics in *Benson* and *Flook*) upheld the subject-matter patentability of certain software-implemented inventions.

Now, the Patent Office again seeks relief from its mandated task of examining patents on software implemented inventions. Confronted by the implications of Commissioner Banner's decision to follow this Court's and the C. C. P. A's rulings and to establish new guidelines for the examination of computer programs, and confronted with the administrative difficulties that such examination may entail, the Patent Office is here again engaged in its Sisyphean task of asking this Court for relief from that inconvenience.

However, the Patent Office can be relieved from its mandate only by the Congress which has set it. As the Court stated in *Diamond v. Chakrabarty*, — U. S. — (June 1980):

> "Our task, rather, is the narrow one of determining what Congress meant by the words it used in the statute; once that is done our powers are exhausted".

Unlike the *Chakrabarty* case, this stored-program technology has been in existence since the 1940's, prior to the 1952 codification of the patent statutes by Congress in 35 USC (see App. A, A14-15, f.n. 60). The stored-programmed machine is an old technology, not a new one, and after more than thirty years of being in the mainstream of the developing "useful arts" should be recognized under the Patent Laws.

The Patent Office suggests (Br. P. 24) copyright protection; but that would be inadequate and inappropriate to the task, and destructive of the constitutional purpose of the patent system as an incentive to machine innovation. There is no evidence of intent of Congress to take away from software inventors the benefits of the patent laws.

Copright protects particular expression, not machine structure. There is no requirement of innovation, only originality; therefore, a copyright would not protect the machine configuration that is at the heart of the stored-program invention. See *Data Cash Systems, Inc. v. JS&A Group, Inc., et al,* 480 F. Supp. 1063 (DC, Ill. 1979), where the court *found as a fact* that "the computer program [stored in a computer] is a mechanical device which is engaged in the computer to become an essential part of the mechanical process"; f.n. 4, ¶ 2; it is therefore, not coprightable subject matter.

The fact that a "program" can be represented as a list of computer operations is no more significant than the fact that a radio circuit can be represented in a schematic diagram. An innovative radio would not be protected by a copyright on a drawing and the firmware-built computer that is respondent's invention would not be protected by a copyright on a program listing. *Data Cash, supra.*

Legislation to amend the copyright laws (Br. P. 24), if it passes Congress, will ultimately be reviewed by this Court to determine whether the mechanical device of a stored program in a computer is a "writing" within Article I, Section 8 of the Constitution. That copyright decision of the Court some years hence, cannot affect the patent decision in these cases.

CONCLUSION.

For the reasons stated above, it is respectfully urged that the writs of certiorari be dismissed. In the alternative, if this Court reaches the merits of the case, the decision of the Court of Customs and Patent Appeals on patentable subject matter should be affirmed.

Respectfully submitted,

MORTON C. JACOBS,
Attorney for Amici ADR and WHITLOW.

APPENDIX A

IN THE

Supreme Court of the United States

OCTOBER TERM, 1977

No. 77-642

LUTRELLE F. PARKER,

ACTING COMMISSIONER OF PATENTS AND TRADEMARKS,
Petitioner,

v.

DALE R. FLOOK,

Respondent.

ON WRIT OF CERTIORARI TO THE UNITED STATES COURT OF CUSTOMS AND PATENT APPEALS

EXCERPTS FROM BRIEF AMICUS CURIAE FOR THE ASSOCIATION OF DATA PROCESSING SERVICE ORGANIZATIONS (ADAPSO)

* * *

(page 16)

II. Special-Purpose Computers Built by Programming a General-Purpose Computer Form Machines and Machine Processes, Hence Patentable Subject Matter.

A. The Technology of Computers and Computer Programs.

The technological facts of the nature of programmed computers are made clear by (a) the computer experts,

(page 17)

(b) the C. C. P. A. findings in other cases, (c) the Patent Office's own findings some years ago, and (d) the Board of Appeals' specific fact finding in the *Johnston Case.*[17] The computer experts explain the facts of the well-known "feature distinguishing the computer from other mechanisms . . . its restructurability. . . . The program restructures the computer," [18] The findings of fact as to this computer technology by the C. C. P. A. in a long line of cases up to the present [19] have held that a general-purpose computer is a "storehouse of parts":

> But once a program has been introduced, the general-purpose digital computer becomes a special-purpose digital computer (i.e., a specific electrical circuit. . . .).[20]

The special-purpose computer "is physically different from the machine without the program" and is patentable as a machine defined as a new combination of elements under 35 U. S. C. § 112.[21] The Patent Office in its Guidelines for Examination of Computer Programs (1966) [22] written in

---

17. See f.n. 24, *infra.*

18. Professor Ivan Flores, City University of New York "Computer Software," *Science & Technology,* No. 89, May, 1969, pp. 16, 17.

19. *In re Prater and Wei,* 415 F. 2d 1393 (1969); *In re Bernhart,* 417 F. 2d 1395 (1969); *In re Comstock,* 481 F. 2d 905 (1973); *In re Knowlton,* 481 F. 2d 1357 (1973).

20. *In re Prater, supra,* f.n. 29.

21. *In re Bernhart, supra,* at 1400, Judge Rich in the *Johnston Case,* 502 F. 2d 765, 773 reaffirmed "the validity of this principle [namely] a new program makes an old general purpose digital computer into a new and different machine." There has always been unanimity and no dissent in the court below on "the validity of this principle."

22. 829 O. G. 441, 442, by Order of Commissioner of Patents Brenner, August 9, 1966.

(page 18)

part by members of the Patent Office Board of Appeals,[23] recognized that the stored program converts that " 'warehouse' of unrelated parts" of the general-purpose computer into a "special-purpose machine" which may be patented as "a novel and useful machine combination due to the interrelation of parts . . . and the fact that portions of the complete machine take the form of a replaceable program is of no moment."

In the *Johnston Case*, the Patent Board of Appeals found:

> . . . that an unprogrammed, or a differently programmed, general-purpose computer is [not] the same machine as the one that would result from appellant's programming of the same or of a similar computer.[24]

The following discussion of the nature of the computer and stored-program technology is presented in order to assist the Court in understanding that respondent's preferred embodiment of a programmed computer for automating an oil refinery's alarm limit control is a "concrete embodiment in a specially devised apparatus," the Patent Office's contrary and unsupported assertion (Br. Pet. 14) notwithstanding.[25]

---

23. *See* Commissioner of Patents Brenner, "The Future of Computer Programs in the United States Patent Office," *Law of Software*, 1968, p. B-8.

24. Reported in *Dann v. Johnston*, No. 74-1033, Petition for Writ of Certiorari, 25A, at 28A.

25. Petitioner's attempt to have this Court determine by legal fiat and contrary to the facts, the nature of computers and computer programs brings to mind the declaration of the Court of the Special Inquisition that the sun rotated around the earth in the face of Galileo's evidence of a different structural relation of sun and earth; *see* J. Bronowski, *The Ascent of Man*, (1973) p. 213, *et seq.*

(page 19)

1. Respondent's Programmed Computer for Automating an Oil Refinery Is a Circuit Configuration.

Respondent's preferred embodiment of a programmed computer is constructed as a configuration of electrical components formed in "a specific electrical circuit".[26] This structure of respondent's machine (and of other machines implemented with a general-purpose digital computer and an internally-stored computer program) is illustrated in the accompanying diagram of the hierarchical structure of a special-purpose computer (see p. 20).

In this diagram, each level of the hierarchy,[27] supplies the building blocks out of which the circuits of the next level of the system[28] are built. The bottom level consists of electrical components from which the logic circuits of the next level are configured. The logic circuits are combined into the more complex building blocks —circuits called the "functional units"[29] of the general-purpose computer's processor.[30] These functional units, in turn, are combined into the special-purpose computer (such as respondent's) which receives input signals (for the values of the oil refinery's "process variables") and pro-

26. *In re Prater, supra,* f.n. 29.

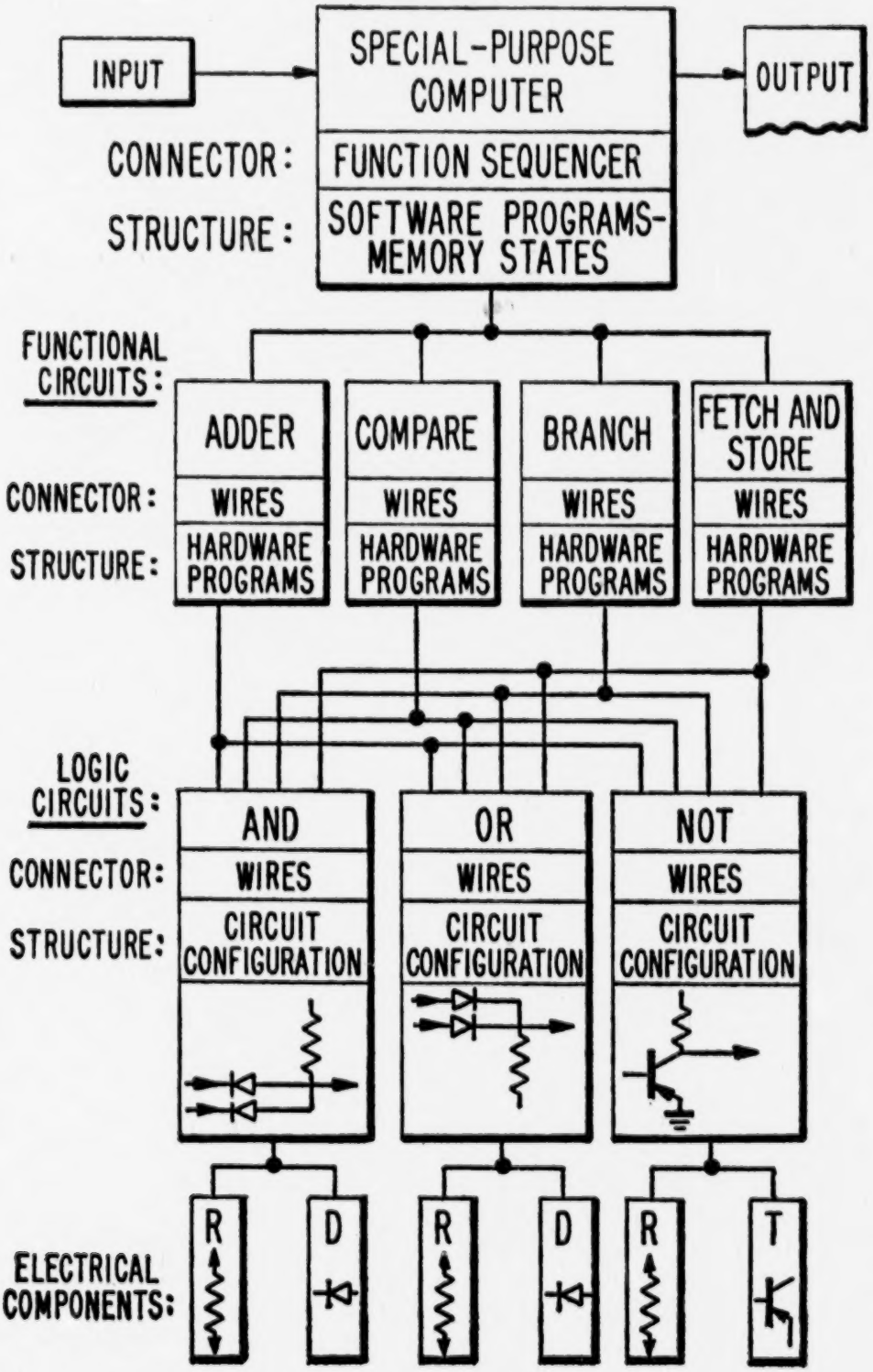
27. "*Hierarchy* . . . a rank arrangement in which the nature of function at each higher level becomes more broadly embracive than at the lower level." (A. Iberall, *Toward a General Science of Viable Systems,* McGraw-Hill, 1972, p. 388.)

28. "*System,* A complex unity formed of many, often diverse, parts subject to a common plan or serving a common purpose. . . ." (Ibid. at p. 7).

29. Professor Ivan Flores, *Computer Logic: The Functional Design of Digital Computers,* (W. L. Everitt, Editor) Prentice-Hall, Inc., 1960, Chapter Eleven, "Functional Units," pp. 173-207.

30. And into the other circuits (*e.g.,* memory, sequencer control, input and output) that form the general-purpose computer. "The term *logic* . . . describes the interrelation among the primary building blocks of the computer." Flores, *ibid.,* p. 123.

(page 20)


INPUT
SPECIAL-PURPOSE COMPUTER
OUTPUT
CONNECTOR:
FUNCTION SEQUENCER
STRUCTURE:
SOFTWARE PROGRAMS-MEMORY STATES
FUNCTIONAL CIRCUITS:
ADDER
COMPARE
BRANCH
FETCH AND STORE
CONNECTOR:
WIRES
WIRES
WIRES
WIRES
STRUCTURE:
HARDWARE PROGRAMS
HARDWARE PROGRAMS
HARDWARE PROGRAMS
HARDWARE PROGRAMS
LOGIC CIRCUITS:
AND
OR
NOT
CONNECTOR:
WIRES
WIRES
WIRES
STRUCTURE:
CIRCUIT CONFIGURATION
CIRCUIT CONFIGURATION
CIRCUIT CONFIGURATION
ELECTRICAL COMPONENTS:
R
D
R
D
R
T

(page 21)

duces output signals (the updated alarm values used to adjust the refinery's alarm limit).

The electrical components, e.g., resistors (R), diodes (D), transistors (T), at the bottom level [31] of the hierarchy are like those used in all modern electronic equipment (e.g., in radar, television, radio, telephone, automation). These electrical components are combined in different circuit configurations to form logic circuits called AND, OR and NOT which transform electrical signals in a manner representative of their named logic operation.[32]

The logic circuits are structured at the next level into functional circuits (Adder, Compare, Fetch and Store, and Branch or Conditional Transfer of Control). These functional circuits combine electrical signals in accordance with their named function. The Adder functions to add numbers represented by combinations of signals. Similarly, the Compare circuit operates to compare numbers for the relations of "equality" and "greater than" and "less than." Fetch and Store are control circuits for moving data signals to and from the memory, which is constructed from millions of flip-flop circuits, the basic storage units for the binary computer signals.[33]

The Branch or Conditional Transfer of Control performs a function unique to digital computers, whereby the sequence of functional circuits can be varied in accordance with different *conditions* of the input or processed

31. At a still lower level of this structural hierarchy, the molecular and atomic substructure of these components (such as the solid-state transistors and diodes) are important factors in their design and in their interrelationships at higher hierarchical levels.

32. The AND circuit produces an active output signal when all of the circuit's inputs are active; OR, when either one or another, or all of its inputs are active; NOT produces an active output when its input is not active. In place of three different types of logic circuits, many digital computers are built from combinations of a single such circuit, the NOR (the negative of OR; its output is active when neither of its inputs are active).

33. The flip-flop is also formed from the logic circuits.

(page 22)

data. In a simple example, different sequences of circuits are activated depending on which of two numbers is the greater.[34]

A general-purpose computer is configured to have a basic "repertoire" of functional circuits (also "instructions" or "orders").[35] The structural configuration of each of these functional, control and memory circuits is called "permanent programming"[36] or "hardware programming."[37]

Programs in the broader sense are part of all automatic machines, and automated systems. "*Program elements* determine *what* the automated system shall do and *how* the parts of the system must function in order to accomplish the desired result."[38]

"Software programming"[39] is the structure of the special-purpose computer which configures the functional circuits (Adder, Compare, Branch, etc.) to interconnect certain sequences of those circuits and combinations of those sequences.[40] The software programming (the in-

34. This Branch control (or "Decision") is fundamental to the ability to structure a general-purpose computer into a special-purpose computer by means of software programming.

35. A more elaborate set (*e.g.*, to perform also subtraction, multiplication and division, shifting of numbers, etc.) is often used.

36. I. Flores (1960), *supra*, p. 42.

37. Prof. Saul Rosen, "Hardware Design Reflecting Software Requirements," 33 *Proc. Fall Joint Computer Conference*, 1968, Pt. II, pp. 1443, 1444.

38. "The program may consist of mechanical parts such as gears, cams, and linkages that connect the various portions of a machine together and determine the timing of the various actions of the machine. In the most modern computer-controlled systems, the program may be stored in the form of punched paper cards, paper tape, magnetic tape, ferrite cores, electronic circuits, or other mechanical or electronic "memory" devices. 2 *New Encyclopedia Britannica: Macropaedia*, 505, 506, "Automation."

39. Term used herein for what is customarily called "computer programming" to avoid semantic confusion with hardware programming.

40. Basic repetitive sequences are called "sub-routines." (Br. Pet. 15).

(page 23)

ternally stored program) takes the physical form of combinations of different states of the memory circuits and of electrical signals that control the states and configurations of the functional circuits.

Software programming resembles hardware programming in a number of respects, and most importantly in the way that lower-level building blocks are structured.[41] However, the connector used with the software for configuring the functional units is a different type from that of the hardware programming. Starting at the bottom level of the hierarchy, as shown in the diagram, the electrical components are connected together in the logic circuits by fixed wire or "hardwire" connectors.[42] The logic circuits are similarly hardwire connected into the functional circuits and other circuits of the general-purpose computer. However, at the top of the hierarchy, a "softwire" connector is employed for the software structure. This connector includes a function ("instruction") sequencer, which effectively interconnects the functional circuits in the sequences prescribed by the software programming.[43]

By forming the structure at the top of the hierarchy as electrical signals, that structure can be stored in the computer memory in the same way that other signals are

41. The software and hardware programming are engineering equivalents, as discussed hereinafter.

42. "Hardwire" connections may be made by simply twisting or soldering wires together or in more advanced fabrication by printed circuits or integrated circuits; *see Scientific American* article, May, 1975, "Microprocessors."

43. The function sequencer activates by control signals the circuitry of the functional units at the appropriate times in sequences specified by the software programming. When activated the circuits are effectively "connected," and when inactive they are not. The engineer's term "softwire" refers to use of control signals for the connections. The sequencer, memory, input and output circuits are all part of the general-purpose computer; the control program is in the states of the memory circuits.

(page 24)

stored. The ability to store the structure of a special-purpose computer in the form of electrical signals gives rise to the ability to replace one such structure by another simply by replacing one set of electrical signals (software program) by another. *This concept of the stored-program computer is the foundation on which the computer industry is built.*

Since the building blocks of each hierarchical level of such computers are nested within those of the next higher level as shown in the diagram, the special-purpose computer at the top level includes all of the elements and structural relations shown in the drawing. This special-purpose computer receives input signals and, within its circuits, combines them and modifies them to produce output signals. The combining and modifying is in accordance with the structure of the stored software programming, which uniquely establishes the "rule of action" [44] of the special-purpose computer.

Respondent's special-purpose computer, in the same way, functions as "a specific electrical circuit" [45] configured out of sequences of functional circuits, which in turn are configured combinations of logic circuits, which in turn are configured combinations of circuit components.

2. The General-Purpose Computer Is an Incomplete Machine That Serves as "a Storehouse of Parts" and Is Completed by Respondent's Stored Program to Form a New Combination of Circuits as Respondent's Programmed Computer for Automating an Oil Refinery.

The general-purpose computer, though extremely complex, is an inchoate, and *incomplete* machine. The functional circuits (Adder, Compare, Branch, etc.) are

44. Robinson on Patents, Vol. I, p. 257, § 173.

45. *In re Prater, supra.*

(page 25)

not interrelated nor do they co-act in any useful way until structured [46] by the software programming into a special-purpose computer. The functional circuits are building blocks used by the software; and in that respect the general purpose computer is a " 'warehouse' of unrelated parts." [47]

This "storehouse of parts" [48] character of the general-purpose computer is evident from the hierarchical diagram on p. 21. Starting at the bottom level, each level serves as a storehouse of parts for the next higher level. The electrical components (resistors, diodes, transistors) serve no purpose by themselves and acquire functions only when connected in the various circuit configurations of the logic circuits. Similarly, the logic circuits serve only as a storehouse of parts for assembling the different circuit configurations of the functional units, and the latter circuits which constitute the general-purpose computer are interconnected by the software programming in the various sequences and combinations that make up the special-purpose computer.

The set of building blocks at each level is similar to the familiar toy Erector Set, whose parts can be connected in an unlimited number of ways to form machines of a wide variety of types. For example, with a storehouse of the electric components (resistors, diodes and transistors, etc.) almost all of the known electrical circuits can be built, including those of the computer logic circuits. With a storehouse of logic circuits, all of the various functional and other circuits of general-purpose computers can be built.[49] Similarly, the function circuits are interconnected

46. *See* Flores, (1969), *supra*, pp. 16, 17.

47. Patent Office Guidelines (1966), *supra*.

48. *In re Prater, supra*.

49. Special-purpose computers are also built directly from the logic circuits using permanent or hardware programming. One could also use the logic circuits as the functional units of the general-purpose

(page 26)

by the software into an endless variety of automatic machines—a musical synthesizer, the controls of a lunar landing module, a document reader for electric meter reports, a calculator for thereafter calculating and printing the associated electric bills. Respondent's programmed computer is similarly constructed in its preferred form of construction.

There is no predetermined relationship of the function circuits (Adder, Compare, Branch, etc. or their individual functions) to any of the above special-purpose computers and other devices; they are "*unrelated* parts" [50] (emphasis added). Nor is there a predetermined relationship of the resistors, diodes and transistors to a computer or television set. However, when the functional circuits of the general-purpose computer are structured by software programming—such as that of any of the above examples—these circuits are organized and interrelated by that programming just as are the electrical components in a television circuit. In that condition, the function sequencer makes the softwire connections called for by the stored program so that the functional circuits are connected into a special-purpose computer having the structure called for by the program.

The hardwire programming is used in building the functional circuits from the logic circuits and softwire programming is used in inter connecting the functional circuits into a special-purpose computer; both connect building blocks into more complex structures.

In short, as the Government asserted in another case, a software program may be "a technologically innovative

---

49. (Cont'd.)

computer, namely, the building blocks for software programming; but "such an approach would be impractically slow." Goldstine, H. H., *The Computer From Pascal to von Neumann*, (1972), p. 257.

50. Patent Office Guidelines (1966), *supra*.

(page 27)

product" [51] and may be demonstrated to have "technological superiority".[52]

The phrase "patents for computer programs" as used by the Patent Office to describe the key issue in these "computer programming" cases,[53] is technically imprecise and misleading in that what is sought to be protected in this technology is not some abstraction unrelated to a machine, but the structure of a particular special-purpose machine or machine-control process which may be produced by the softwired or programmed general-purpose computer but which also may be hardwired.

3. Hardware and Software Programming are Equivalent and Alternative Approaches for the Construction of Computerized Control Systems and Equally Patentable.

The Patent Office has set up a false dichotomy in the patentability of computerized control systems. Those implemented by hardware programming (*e.g.*, the general-purpose digital computer) are recognized to be machines by petitioner and patents covering such machines and their control processes are granted subject to their meeting the statutory tests of novelty (35 U. S. C. § 102) and unobviousness (35 U. S. C. § 103). However, computers implemented by software programming are generally denied patentability.[54]

---

51. Government's Pretrial Brief in *United States v. IBM*, 69 Civ. 200, at p. 202.

52. *Ibid.* at p. 29. This Government Brief in the IBM case makes it clear that the software industry is in the business of selling technological products (machines) not "abstract ideas."

53. *Benson, Johnston, Chatfield,* and *Noll,* the predecessor cases.

54. The Patent Office is not consistent in this respect; many of the Examiners understand the machine structure of software, as shown by the patent for an "Analytic Character Recognition System" implemented by software, U. S. 3,873,972.

(page 28)

Petitioner's distinction between the two alternative forms of computer construction is inconsistent with the well-established technological "principle of the equivalence of [software] programming and hardware. This principle [is based on the fact] that a program for a general-purpose machine turns it into a special-purpose machine." [55]

Petitioner has acknowledged in the previous *Johnston Case* this equivalence of hardware and software, namely, that a special-purpose computer can be built either by hard-wired logic circuitry or by programming a general-pupose computer.[56] The particular means that is used depends upon many business and engineering trade-offs.[57]

---

55. "The principle becomes more tangible when we compare in detail the logical design of that special-purpose machine with the flow chart of that program." Prof. Saul Gorn, Final Report No. AD 59 URI, at p. 4, from Univ. of Penna. Moore School of Elec. Eng. to U. S. Army Signal Research & Development Labs, 1959.

56. "The particular operations to be performed by a computer are governed by the state and configuration of its electronic circuitry [which] can be fully built into the computer through wires. . . . Alternatively, the computer may be designed so that the state or configuration of its circuitry can readily be changed without rewiring or other visible physical changes . . . by use of different 'programs.'" (See *Dann v. Johnston* No. 74-1033, Brief for the Petitioner at 14-15.)

57. "Perhaps the most important part of the job of computer system design is to determine which functions are best handled by hardware, and which should be provided by software. All of the traditional engineering considerations come into play here. One must consider the added cost of additional hardware measured against possible gains in operating efficiency. One must consider the availability of manpower with the appropriate skills for design and for implementation. One must also consider intangible factors such as the inherent advantages of simplicity and the hidden costs of complexity.

"From the point of view of the computer user it makes no difference which features are incorporated into the hardware and which are programmed in software.

"It is the combined hardware-software system, the 'extended machine' that the user must rely on to provide the power and flexibility and convenience that have been promised with each new generation of computing equipment." S. Rosen (1968), *supra*.

(page 29)

None of the factors concerning the choice of hard-or-softwiring has any relation to inventiveness of the underlying machine. If the circuit states and configuration of a particular special-purpose computer contain the elements of invention, the machine is patentable and the fact that the wiring is "hard" or "soft" is immaterial, for the two types of computer construction are interchangeable.

This equivalence is inherent in the concept of the stored-program computer. The first electronic computer, the ENIAC built about 1945, was "not a stored-program computer. Programs were installed and changed by engineers who changed the wiring among its various components."[58] The stored-program concept, invented while the ENIAC was being built, had as its very purpose the replacement of the hardwire connections by the softwire. The first published description[59] of the stored-program computer clearly explained (p. 1, ¶ 1.2) that the program ("instructions"[60]) was "an integral part of the [computer's] structure."

---

58. S. Rosen, Purdue University, "Electronic Computers: A Historical Survey," *Computing Surveys*, Vol. 1 (1969), pp. 7, 8.

59. A. W. Burks, H. H. Goldstine, J. von Neumann, *Preliminary Discussion of the Logical Design of an Electronic Computing Instrument*, 1946. In this famous "paper that became a landmark in the history of computer sciences if not in the history of human thought," (4 *New Encyclopedia Britannica: Macropaedia*, 1045, 1047) John von Neumann and his colleagues at the Institute for Advanced Study in Princeton first published "this stored-program concept" on which the modern computer is based and the technology of which underlies the issues in this case.

60. A human receives *instructions* on how to operate the manual choke in an automobile. When an *automatic* choke is built in by means of a "program," we don't think of "instructing" the engine or its controls. Yet that is the terminology of the computer industry established thirty years or more ago to describe the automatic controls of a stored program. The automatic control of the choke in future cars will be done by a more advanced programmed general-purpose computer, and appropriate computer "instructions" will be used for that purpose. *See* Vacroux, *infra*, f.n. 76, at p. 40.

(page 30)

As far back as 1949, when the first stored-program computer was constructed in England, its builders recognized the equivalence of programming by hardwire and softwire, and explained that additional functional circuits could be added to the computer or the equivalent software programming could be used to implement the same functions.[61]

The Patent Office has no difficulty in finding patentable subject matter in the hardware approach to building special-purpose computers which technologically began to be replaced 30 years ago with stored-program inventions. However, it insists that the stored program approach which is the present foundation of the computer industry and is revolutionizing the entire electronics industry is not patentable. To base disparate legal treatment upon a mode of construction has no justification in law, technology or common sense. Softwired (*i.e.*, wireless) programming is an historic advance over the wired program. The patent statutes and their constitutional base were not intended to be limited to a finite period of technological development. There is no justification for the Patent Office's position that technology has now outstripped the boundaries of patent law.

The equivalence of hardware and software underlies this revolutionary change taking place today in the electronics industry. Programmed microcomputers (general-purpose computers constructed with "microscopic components"[62]) are used to build electronic apparatus in "a wide range of applications from specialized laboratory instruments to sales terminals and electronic games."[63]

---

61. M. V. Wilkes, "Progress in High-Speed Calculating Machine Design," *Nature*, Vol. 164 (1949) pp. 341-3.

62. A. G. Vacroux, "Microcomputers," *supra*, pp. 32, 33.

63. *Ibid.*, p. 34.

(page 31)

"Until recently digital circuits were designed almost exclusively with 'hardwired' logic . . . although this approach is very powerful (indeed, it is used in most of the electronic equipment manufactured today), it yields an *inflexible* product that is difficult to modify or improve without major redesign. Most of the microcomputers in service today were adopted as a replacement for *inflexible* circuitry. . . . *it is to their flexibility, inherent in all stored-program systems, that they owe most of their popularity.*[64]

Thus, the general-purpose computer has become (in miniature size) a tool that is structured by the addition of programs to build all kinds of instruments and apparatus. Each progammed microcomputer is a different electronic machine as is each hardwired device that the programmed microcomputer is replacing. Such electronic machines have unquestionably been patentable subject matter under 35 U. S. C. § 101 when hardware programming was used. The objection to patentability based upon the use of the more modern and "flexible" technology of the stored program cannot be justified by any basis in law or fact.[65]

The denial of patents to a modern technology (the softwire mode of building special-purpose computers and other machines) raises anew the question: whether the

64. *Ibid.*, p. 39.

65. The failure to recognize the equivalence of hardware and software opens a pandora's box and leads inevitably to a "raiding" situation in which existing hardware patents have no legal protection from infringing software-constructed devices. *See Digitronics Corp. v. The New York Racing Association*, (E. D. N. Y.), 187 U. S. P. Q. 602, affd. on other grounds 553 F. 2d 740, cert. dnd., — U. S. —. This problem is not limited to the computer field, for many automatic machines have the potential to employ a programmed general-purpose computer to replace their existing electronic, electrical or mechanical controls. *See* 2 *New Encyclopedia Britannica, "Automation," supra.*

(page 32)

Patent Office "freeze[s] . . . patents to old technologies, leaving no room for the revelations of the new, onrushing technology." (*Benson, supra,* at 71.)

4. Software Is Not an "Abstract Idea" and Its Construction Is Engineering.

The software products segment of the computer industry is composed of hundreds of small and medium size companies, many of which are members of ADAPSO. The programmers build software out of building blocks to engineer the special-purpose computer in its hierarchical structure,[66] much the same as that required for other modern complex machines.

In the hierarchy of computer systems, the logic circuit designer builds logic circuits such as "AND" and "OR" circuits without concern for the molecular structure of the individual transistors and other components that make up those electronic circuits (which concern is left to component designers working at a lower level of hierarchy). The designer of the hardware function circuits such as "ADD," "COMPARE" and "STORE," configures logic circuits without concern for the detailed structure of the individual logic circuits he uses to construct the hardware function circuits. Similarly, the "programmer," the designer of the special-purpose computer, configures hardware function circuits into the structure of the ultimate special-purpose machine without concern for the detailed structure of the individual hardware function circuits. The same hierarchical process is followed if, instead of softwire programming, hardwire programming is used to build the special-purpose computer.

66. See "Hierarchies–Software Engineering Design," Software Engineering, International Computer State of the Art Report, 1972, p. 47 et seq.

(page 33)

These building blocks lie on successive levels of a hierarchy of circuit construction and the software engineering is performed at the highest level. The software itself, because of its complexity, often must also be constructed in a number of hierarchial levels, and the software engineer should preferably work like:

> ". . . the electrical engineer designing a new T.V. set [who] certainly doesn't design each vacuum tube, transistor, and capacitor all over again; he relies on existing components. In fact, he normally designs on a much higher level, using 'off-the-shelf' power supplies, oscillators, etc." [67]

A recent IBM report further demonstrates the similarities in software and hardware engineering:

> Structured programming reduces the arrangement of the program logic to a process like that found in [hardware] engineering where logic circuits are constructed from a basic set of figures. As such, it represents a standard based on a solid theoretical foundation.[68]

Working at different levels in the system hierarchy of the special-purpose computer to be built, contemporary software engineers as that IBM report explains, may use a "structured programming" [69] approach similar to the design of other automatic machines:

> The top-down approach is patterned after the natural approach to system design and requires that

67. The IBM report, "Composite Design: The Design of Modular Programs," Jan. 1973, p. 6.

68. "Management Concepts for Top-Down Structured Programming," IBM Report, February 1973, p. 1.

69. *Ibid.*, Abstract.

(page 34)

> programming proceed from developing the control architecture (interface) statements and initial data definitions downward to developing and integrating the functional units.

In the Apollo Space Shuttle Software System, a mammoth software development effort:

> . . . program structuring is of major importance; decisions . . . determine the total structure of the software both statically and dynamically. Program structuring defines the building blocks, the control mechanisms and the interfaces of the software. It is in this effort that the interrelationship of systems software, applications software and data is entirely mapped out.[70]

The software engineer builds complex automatic machines as the foregoing examples amply demonstrate. Neither the software engineer nor the hardware engineer devises "abstract ideas" or "mathematical algorithms," contrary to the Patent Office's suggestions (Br. Pet. 17) and to what CBEMA would have the Court believe (Br. CBEMA 17).

The design and development process in this technology is quite similar to that of any other technology. First the goal is decided upon, then the prior art is explored for possible solutions and, if not found, the "technological roadblocks" are identified.[71] Invention may take place in overcoming these roadblocks.

Parallels to the design and development process of other machines may be seen in the software design process

70. "Top-Down, Bottom-Up, Structured Programming & Program Structure," Draper Labs, MIT, Dec. 1972, pp. 28, 31.

71. *See, e.g., Computers in the 1980's,* R. Turn, Columbia Univ. Press, 1974, p. 21.

(page 35)

presented in a paper at the World NATO Conference entitled "Software Engineering" [72] which took place in Germany in 1969 and was attended by over 50 experts and prominent leaders in the software field.

° ° °

---

72. "Software Engineering," Report on a World Conference Sponsored by the NATO Science Committee, Garmisch, Germany, Jan., 1969, p. 13.

APPENDIX B.

LAW OFFICES
MILLMAN AND JACOBS
PATENT, TRADE MARK AND COPYRIGHT CAUSES
1420 WALNUT STREET, SUITE 1420
PHILADELPHIA, PENNSYLVANIA 19102

MORTON C. JACOBS
MAX R. MILLMAN

PHILADELPHIA (215) KI 6-5950
NEW JERSEY (609) 964-5588

January 26, 1979

The Honorable Jordan J. Baruch
Assistant Secretary of Commerce
for Science & Technology
U. S. Department of Commerce
Washington, D. C. 20230

Dear Dr. Baruch:

Your talk last week before the Philadelphia Patent Law Association on the role of the patent system in meeting our nation's need for technological innovation was most encouraging.

I was pleased to chat with you and to receive your favorable response to our request for a meeting with ADAPSO, the software association, which we represent as patent counsel and which is much concerned with stimulating innovation in computer technology and particularly in computer software.

The ADAPSO Software Protection Committee, in requesting a meeting with you, seeks to discuss how to facilitate for software innovators their rightful access to the incentives of the patent system.

ADAPSO's position is that patents should be granted for software-embodied inventions equally with those granted for hardware-embodied inventions. A choice between these two modes of its constuction should not make any difference in the patentability of an invention.

ADAPSO's view is based on the technological principle of the engineering equivalence of software and hardware—a principle which you readily acknowledged in response to a question after your talk.

To Hon. Jordan J. Baruch Page 2

January 26, 1979

In view of your grasp of the underlying technology, your direct participation is needed in answering this momentous and long-standing question, which the U. S. Supreme Court did not deal with in three "computer programming" decisions.

We would like to suggest a date for the meeting in March or April 1979 convenient to your calendar.

At your direction, we are concurrently channeling this request to the Honorable Joseph F. Nakamura, Patent Office Solicitor, who, you suggested, would participate. We concur (having worked with him for tens of years) in the high regard that you expressed for Mr. Nakamura.

A copy is also directed to the Commissioner of Patents for his information.

Sincerely yours,
Morton C. Jacobs

MCJ:swh

Copy: Honorable Donald W. Banner
Commissioner of Patents and Trademarks

Honorable Joseph F. Nakamura
Solicitor, U. S. Patent and Trademark Office

Mr. Bruce T. Coleman, Vice-President, ADAPSO;
Group Vice-President Software Products
Informatics, Inc.
Woodland Hills, California

Mr. Martin A. Goetz, Chairman
ADAPSO Software Protection Committee;
Senior Vice-President, Applied Data Research, Inc.
Princeton, New Jersey

Mr. Jerome Dreyer, Executive Director, ADAPSO
Arlington, Virginia

Milton R. Wessel, Esq., ADAPSO Counsel
Pleasantville, New York

LAW OFFICES
MILLMAN AND JACOBS
PATENT, TRADE MARK AND COPYRIGHT CAUSES
1420 WALNUT STREET, SUITE 1420
PHILADELPHIA, PENNSYLVANIA 19102

MORTON C. JACOBS
MAX R. MILLMAN

PHILADELPHIA (215) KI 6-5950
NEW JERSEY (609) 964-5588

April 30, 1979

The Honorable Jordan J. Baruch
Assistant Secretary of Commerce
for Science & Technology
U. S. Department of Commerce
Washington, D. C. 20230

Dear Dr. Baruch:

We are pleased with the meeting that ADAPSO held with you on April 3, 1979 to discuss patent protection for software. We were also pleased to meet at the same time with the Honorable Donald W. Banner, Commissioner of Patents and Trademarks, and the Honorable Joseph F. Nakamura, Solicitor, U. S. Patent and Trademark Office.

As you suggested, we can quickly "stipulate" that there is a need for software innovation in the computer industry and that there is a small business potential to innovate in the software field. There was also agreement on the importance of the Patent System to stimulate technological innovation.

We keenly appreciate that your expertise in this software technology enabled us to get to the essential issues. As you noted, the decision to build a machine in hardware or software is one based on engineering factors such as relative cost. Thus, these two forms of construction (known also as "hardwire" or "softwire") are engineering equivalents, a principle that has been understood from the beginnings of this technology some thirty years ago.

For that reason, and since the mode of constructing an invention (e.g., be it hardwired or softwired) can make no difference in its patentability, ADAPSO has for ten years and more been urging that the patent protection that has always been available for hardware-implemented inventions should likewise be available for the same invention that is software-implemented.

This should be plainly evident from technological considerations, for the form of an invention that is first built (for example, in the computer field) is frequently softwired. In patent law, the first form of an invention is also particularly significant, and often favored, for a variety of reasons.

The Supreme Court's decision in the five so-called "computer programming" cases that have been appealed reached no findings about the software technology that would suggest a different treatment for a software-embodied invention than for a hardware embodiment.

The Supreme Court's decisions that mathematics as such is not patentable are not contested by ADAPSO. It is our understanding that the Patent Office does not take the position that because an invention is describable in mathematical terms it is unpatentable. Though hardwired computer systems can be described by Boolean algebra, the Patent Office nevertheless issues patents for such systems.

The Court of Customs and Patent Appeals has felt qualified to deal with the technological nature of software from the very first case that came before it in 1969. In

that case, *In re Prater,* the Court concluded than an invention built with software would be patentable subject matter by the nature of the technology:

> . . . a general-purpose digital computer may be regarded as but a storeroom of parts and/or electrical components. But once a program has been introduced, the general-purpose digital computer becomes a special-purpose digital computer (i.e., a specific electrical circuit with or without electro-mechanical components) which, along with the process by which it operates, may be patented subject, of course, to the requirements of novelty, utility and non-obviousness.

The Patent Appeals Court has reaffirmed this technological principle numerous times over the years. Most

recently, in a December, 1978 decision *In re Johnson,* et al, the Patent Appeals Court continues to reject the false dichotomy between software-built and hardware-built inventions:

> We stated in In re Chatfield . . . that "the mere labeling of an invention as 'a computer program' does not aid in decision making" and the Supreme Court declined to decide either Benson or Flook on such broad, nonsubstantive grounds. . . .
>
> Very simply, our decision today recognizes that modern technology has fostered a class of inventions which are most accurately described as computer-implemented processes. Such processes are encompassed within 35 USC 101 under the same principles

> as other machine-implemented processes, subject to judicially determined exceptions, inter alia, mathematical formulas, methods of calculation, and mere ideas. The overbroad analysis of the PTO errs in failing to differentiate between a computer program, i.e., sets of instructions within a computer, and computer-implemented processes wherein a computer or other automated machine performs . . . the process. . . . This distinction must not be overlooked because there is no reason for treating a computer differently from any other apparatus employed to perform a . . . process. . . .

The Patent Office is not bound by an administrative decision taken ten years ago to rule out patents for software-built inventions, particularly where it is based on a technologically unsound premise, which the Patent Appeals Court has specifically rejected and the Supreme Court has declined to deal with. No change in the law is required, neither in the courts nor in Congress (the attempt ten years ago to have Congress legislate this rule of no-patents-for-software did not succeed).

To Hon. Jordan J. Baruch April 30, 1979

ADAPSO believes that the change that must be made is for the Patent Office to stop denying the technological principle of the engineering equivalence of hardware and software—to stop denying that software and hardware are but two different modes of machine construction. This error in how technology is viewed can be readily corrected.

Once that change is made, so that computer inventions are not treated differently as to patentability because of the mode of construction, the incentives of the Patent System can operate to stimulate innovation in software technology.

Sincerely yours,

MORTON C. JACOBS
Morton C. Jacobs
ADAPSO Patent Counsel

MCJ:swh

Copies: Honorable Donald W. Banner
Commissioner of Patents and Trademarks

Honorable Joseph F. Nakamura
Solicitor, U. S. Patent and Trademark Office

Mr. Bruct T. Coleman, Vice-President, ADAPSO
Group Vice-President Software Products
Informatics, Inc.
Woodland Hills, California

Mr. Martin A. Goetz, Chairman
ADAPSO Software Protection Committee;
Senior Vice-President, Applied Data
Research, Inc.
Princeton, New Jersey

Mr. Jerome Dreyer, Executive Director,
ADAPSO
Arlington, Virginia

Mr. Aso Tavitian, President
Whitlow Computer Services, Inc.

Milton R. Wessel, Esq., ADAPSO Counsel
Pleasantville, New York

(Letterhead of)

UNITED STATES DEPARTMENT OF COMMERCE
THE ASSISTANT SECRETARY OF SCIENCE AND TECHNOLOGY
Washington, D. C. 20230

(202) 377-3111 5/28/79

MAY 25 1979

Mr. Morton C. Jacobs
1420 Walnut Street
Suite 1420
Philadelphia, Pennsylvania 19102

Dear Mr. Jacobs:

Thank you for your letter of April 30, 1979 following up on our meeting of April 3 at which we discussed patent protection for software.

We indeed had a very useful exchange of ideas. As you point out, we can quickly "stipulate" to several things. There is a need for software innovation in the computer industry and there is a small business potential to innovate in the software field. We agree too that the patent system is important in stimulating technological innovation.

I very much appreciate ADAPSO's efforts to help clarify the issues with respect to this topic which has such important economic implications.

Sincerely,

JORDAN J. BARUCH
Jordan J. Baruch